International Journal of Software Engineering and Its Application
Vol.11, No.7 (2017), pp.31-42
http://dx.doi.org/10.14257/ijseia.2017.11.7.03

News Real Time Recommendation Framework for Websites Contents

Jalal Omer Atoum[1*] and Ibrahim Mohamed Yakti[2]

[1]Southern Arkansas University, Magnolia, Arkansas-USA
[2]Princess Sumaya University for Technology-Amman-Jordan
[1]jalalomer@saumag.edu, [2]ibrahim.yakti@gmail.com

Abstract

The expeditious magnification of technology in both software and hardware resulted in moving several industries such as media, news, publishing, and printing from classic approach to more digital approach. The experience of recommending products based on user's behavior showed huge impact on businesses. Many studies were done on batch processing and showed that the bottleneck in recommendation algorithms is the search for neighbors among a large user population of potential neighbors. This paper proposes a framework for recommending content for news websites to users in real time to increase both user and business satisfactions using academic and news industry standards, it starts with gathering data and ending with delivering personalized recommendations per user. Results showed an improvement of users engagement when the recommendation was active; average user time was increased by 39.45%, while the users engagement with the recommendation feature was 22.9% of the users.

Keywords: *Web News Recommendation Systems, Content-Based Filtering. Collaborative Filtering, Website Users Engagement.*

1. Introduction

News delivery has been changed from classic delivery methods where news providers publish their news offline like newspapers or broadcast their programs over TV or radio to more innovative methods using modern technologies such as social media services, podcasts, newsletters, blogs, and news websites. Moreover, the size of the internet is expanding daily; the number of web-pages available on the internet had exceeded 47.8 billion web-pages [1]. When readers want specific information they usually do a search query either on search engine websites or on the specific website they are browsing. The traditional search or listing returns the same results to all users. Also, most of the time readers don not know exactly what they are looking for, especially with news websites. They are only looking for something that might be of their interest, therefore, many processes were developed to increase readers reach by improving website visibility to search engines in process called Search Engine Optimization (SEO) that helps readers to reach a website through search engines. Furthermore, there are many processes that are used to improve readers engagement on a website, to increase number of page-views per visit, time per visit, and click-through rate (CTR) which is the number of links the readers visit divided by the total number of page views, and to reduce bounce rate where readers exit after visiting the first page on the website.

In content websites like news, the value of a homepage is decreasing; as more traffic comes from searches and social media, the homepage as the entryway into a sites content is becoming increasingly obsolete [2]. Users behavior is shifting from surfing the

[*] Corresponding Author

ISSN: 1738-9984 IJSEIA

homepage to explore more into articles pages. In a leaked New York Times report stated that in 2014 only a third of our readers ever visited it. Furthermore, the behavior and the feedback of users should be directly inserted in their learning algorithms and adapted in real time. This is especially important because research has shown that user behavior is highly dependent on daytime and speed [3].

Recommendation systems empowered a lot of online products such as e-commerce, communication, listening to music, reading news and movie recommendations; they reduce the complexity to find relevant information for users.

The main goal of this research is to build and test a real time recommendation framework for web content using industry tools and standards such as google analytics [4] and web events tracking. This framework will be universal as its components can be changed or altered based on the business requirements. It is implemented and evaluated to predict what other articles may the reader dynamically like in real time to add a valued process to the abovementioned processes to give readers recommendations of their interests based on their past interactions in addition to provide publishers the ability to increase their readers' engagement. Finally, this framework will be tested using both business and academic metrics to make sure it has met its purposes.

This paper is structured as follows; Section two presents a detail background and related work required to know about recommendation systems and techniques, problems they face and features that can be used when building the proposed framework. Section three presents in details the proposed framework for implementing a real time recommendation system for news websites. Section four discusses the conducted experiment, their evaluations, and the results. Finally, section five summarizes the conclusions and discusses potential future work.

2. Background and Related Work

Recommendation systems are very useful for creating personalized user experience in many domains such as movie recommendation, book recommendation, communities, social media, news articles, and jobs. There are some successful services that are based on recommendation systems such as Last.fm [5] and Pandora [6]. Also, recommendation systems and techniques are being used for many years, most of these techniques never concerned about the performance with high speed data streams. Studies have been interested in making batch processing that cope with data streams, incremental batch approaches, or in comparison of recommendation algorithms and their limitations using batch and online processing [7].

Two main approaches are used to build a recommendation system: Collaborative Filtering (CF) and Content-Based [8][9][10]. Collaborative filtering recommendation systems use the behavior of other users to predict the behavior of current users, while content-based recommenders use the item's features to predict similar items. Last.fm is an example of collaborative filtering algorithms implementation to predict the interest of the user while Pandora is an example of content-based implementation to predict the user's interest. Moderns recommendation systems use a combined implementation of the abovementioned approaches to solve issues related to each approach, such implementation is called Hybrid recommendation system [11].

Recommendation algorithms are very resource consuming; they are evaluated based on their performance and accuracy. To have better algorithms, we must overcome the challenges that face them, which mainly are; Data (input or output) is said to be challenging when it is of high Volume, Velocity, or Variety [12]. The data generation (volume) in the world is expanding very fast and based on McKinsey Global Institute that had estimated the data volume is growing 40% per year, and will grow 44 times between 2009 and 2020 [13]. Another challenge is that the meaning of some data is variable and may change over time. This last challenge will be very hard especially with algorithms that use language processing, words' meaning may change over time and in different context, for example a position name such as (CEO of Company) may refer to a specific person at the current time while it may be refer to another person in the past or future time [14].

Moreover, any recommendation algorithm is ranked based on its accuracy which is the value returned by that algorithm, having input with all these challenges mentioned above will be very inspiring for an algorithm to tell which part of the data is valuable and which part is not. Systems that use recommendation systems usually expect them to return high quality results in short time. New real world systems require results to be calculated in real time to use the results with the current system's user. The interest prediction of users is a challenging task [15].

This research focuses on recommendation systems that are built on top of systems with two main elements: Items and Users. Recommendation algorithms usually start with either an item then search for similar items, or start with the user and try to search for users who have common attributes with the current user, after that they eliminate shared attributes then return results back to users.

3. Proposed Framework

News content is usually short-life-span content, once the user read the article or once the life-span of the article is expired it is rarely that the user goes back to the article again later. This is different from movies and music products as they have long-life-span and users might repeat the same track, playlist, or movie over and over for a long time span, even come back again to listen to the same tracks after a decade for example. Hence, as news is short-life-span content, the recommendation system cannot be built offline using batch processing and carry the updates for each news article as they will be useless then.

Furthermore, as news articles do not have an inventory or expiry time, they will not run out of stock regardless of the number of reads or views, unlike e-commerce items which they might expire, run out of stock, or have their prices changed, the aim is to make recommendations for users based on their past history compared to similar users' behavior so they stay in stable state with their predicted behavior and that's what the name indicates.

3.1 Test Environment

For the purpose of this research, a test website in Arabic was built on a domain called ("http://www.nakshat.net/") using an Open Source Content Management System (CMS) called WordPress [16] based on PHP [17] and MariaDB [18] database. Test content was uploaded to the website; 90 articles at the beginning of the experiment, one article was added later to test how the framework reacts with new content changes; 23 Technical articles, 20 Sport articles, 17 Cars articles, 8 Animal's world articles, and 53 Entertainment articles. Each article has at least one image, 37 articles come with at least one YouTube video in the body of the article, and all of these articles have text bodies in Arabic.

The main software components used in the test are: Nginx (version 1.8) [19], MariaDB (version 5.5.41), PHP-FPM (version 5.4.16), WordPress (version 4.2.1) [16], Solr (version 5.10) [20], NoSQL features, Solarium (version 3.0) [21], MongoDB (version 2.6.9) [22], Httperf [23], and Google Analytics [4].

A dedicated server [24] is used for all modules of this test, a single server or a scale of multiple servers can be used to implement the proposed framework because of the modular design of the framework.

Finally, this framework is designed to use a service-oriented architecture (SOA) design pattern and Application Programming Interface (API) calls. The focus on the recommendation system components to meet the framework requirements is clear using these methodologies. Such main components can be easily replaced with similar functionality components, many alternatives of each component can be found as free or enterprise solution, therefore, the framework can be integrated with any new or already running infrastructure.

3.2 Framework Modules

The proposed recommendation system consists of three main modules as follows:

1) Data Collection: Studies showed that users usually do not really know what they are looking for and when they do; they are not able to explain their choices, therefore, it is easier to find similar users by observing them and predicting the behavior of one user rather than trying to understand that user. Data collection is the first step of the recommendation system, it is building the users' rating of articles based on their views. Users' views will be used as weights or ranking factors for producing recommendations, i.e.: the more common articles between users, the more weighted and confident the recommendations can be. Data collection is very transparent and done in the background after loading the article page. Figure 1 explains how this module operates in a real production environment.

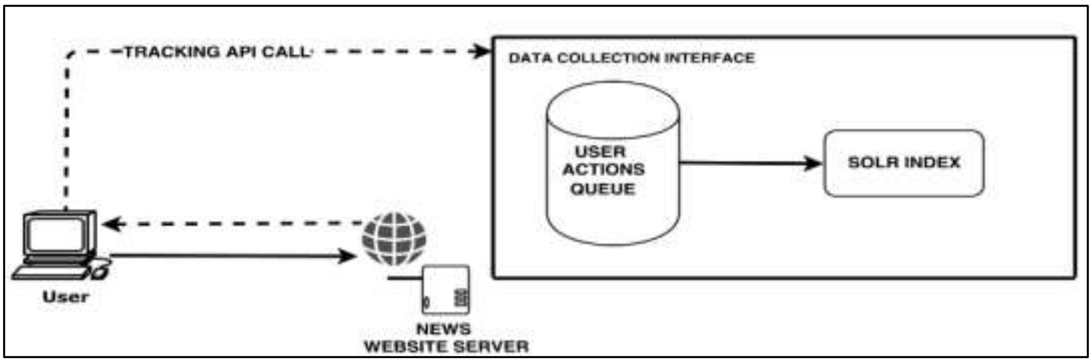

Figure 1. Data Collection Module

The tracking service is called using a Javascript module that is loaded after loading the page, this has a lot of value to the website, the data gathering module, and the recommendation system for the following reasons:

i) A Javascript call is done after page loading, this way it will be guaranteed that the tracking code won't affect the website performance.
ii) If the website uses caching to speed up page load, Javascript calls won't be cached, therefore, it will be guaranteed that the hit will be tracked and it won't require any change to the current website processes.
iii) Javascript calls only run on the client browser, in this way we will not have noise data from search crawlers or any web bots that load the page.
iv) Javascript is cross platform scripting language.

2) Indexing Data: Once the call reaches the server, it gets into a queue and saved on the MongoDB, then it is moved into Solr indexing engine. Indexing data and writing it to a disk is a very expensive process, this is the data incremental process mentioned earlier, so it is done on time basis, based on the content of the test website. It has been noticed that on average the user needs about 2 minutes to view each article, we tried a threshold of 1 minute to move data from the queue in batches into the indexing engine, every minute a job reads all new data in the queue and sends them to the indexing engine. Eventually, the data will be indexed, ranked and be ready before the user finishes reading the article. When the user is moved to a new article, a new recommendation will be ready and waiting for users based on their past actions on the website. For each record, the following data is saved and required for the indexing engine:

i) Unique id: an identifier for the record.
ii) Session id: a unique identifier for each user, the session id is generated automatically the first time the tracking system is called and saved in the browser's cookie.
iii) Article id: a unique identifier for the article generated from the source website.
iv) Property id: a unique identifier to distinguish between properties in case of using the same system to different websites/channels.
v) Timestamp: Date and Time of the article view.

Also, the Solr (the search engine) uses a data structure called "Inverted Index" to save, query, and retrieve data. It helps the recommendation system to get similar sessions and items easily then as the main queries are searching for sessions with specific items, then searching for items with specific sessions excluding the user's items;

3) Calling Recommendations: Recommendations are produced dynamically in real time, each user (session id) will have personalized recommendations based on her/his historical actions. In case of the session is new, popular, and trending articles can be used then as recommendations. Recommendations are generated using Collaborative Filtering techniques. It parses data from users' views and performs the following steps:

i) Get items read by the current user in the past.
ii) Start with similar users:
iii) Find similar users with similar items.
iv) Find most ranked items in similar sessions excluding users' items.
v) If no users with similar items, then find most ranked items among all sessions.
vi) Return ranked recommendations.

The calculation of the rank is based on the user history and items ranking in similar sessions. These ranks are used as weights to improve the recommendation results. Figure 2 shows the full recommendation process.

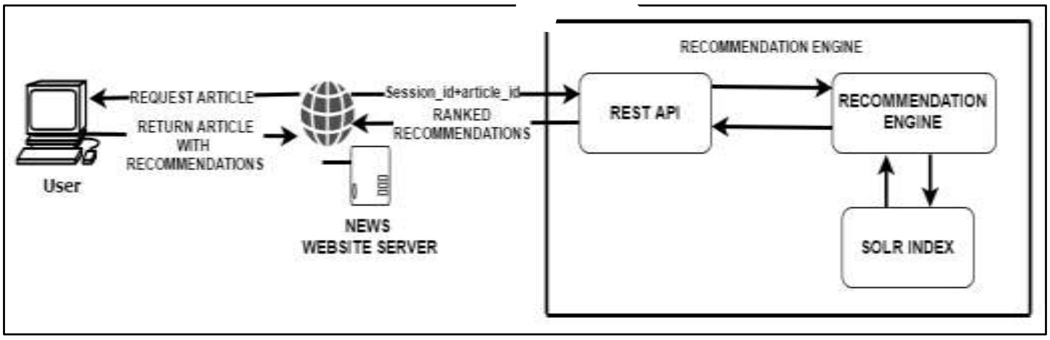

Figure 2. Recommendation System Process

4. Experiments and Analysis

4.1 Framework Evaluation Criteria

Since users usually don't know exactly what they are looking for, it is not enough just to deliver accurate recommendations, but we have to consider that users would like to discover new items that they might not be able to reach by traditional search or browsing. Moreover, considering the high news item lifetime regarding the user, the system should be dynamic and consider items that have been read by the user so not to recommend them again. Our framework is based on real users' behavior on the website, therefore, it is required to measure the change in users' behavior when interacting with the real time recommendation system. In this way, the true value of the system will be shown when it is doing real tasks and normal daily operations on the website such as adding new contents and modifying homepage blocks and content.

Industry standard tools were used to measure the quality of recommendation systems such as Google analytics that; tracks the insights to show the impact of the recommendation system product on the website, shows exactly how people are using the website and interacts with the recommendation system, and meets the framework requirement to measure the users' engagement on the website when the recommendation system is integrated and running.

4.2 Performance

The Httperf tool is used to measure the performance of the recommendations API, it used the recommendations log to simulate the calls. This tool sent 2815 different recommendations requests to the recommendation system, it was able to establish 2815 connections and receive 2815 replies successfully. This means the success rate was 100% and there was no error occurred at any part of the calls, the network is stable. Also, it was able to handle such number of calls in a total of 11.9 seconds, which equals to 236 connections per second. The average connection time was 4.2 milliseconds, the lowest connection time was 0.1 millisecond, and the maximum connection time was 11 milliseconds. On average the benchmarks succeeded in receiving 236.6 replies per second, the lowest rate was 234.6 replies per second, while the maximum was 238.6 replies per second.

4.3 Visits and Page Views Analysis

Web content websites use multiple channels to gain traffic to their websites like online advertisements, referring services, and social share. Facebook ads campaigns is used in our framework to get users on to promote website links and posts. A Facebook page was created to use it to share new posts to page fans. It had been noticed that Social/Organic reach had increased once users started to interact with content on social media and on the test website.

The Facebook campaigns were running in the period from April 25th, 2015 to May 3rd, 2015. They promoted the test website homepage, some articles, and some Facebook page posts. The campaigns ads shown on Facebook were 6.76 Million times (impressions) regardless of their mobile apps, Facebook interface, news feed, or right column area. These impressions reached 530117 people with an average of 12.76 impressions per person. Almost, 4950 clicks (visits) were gained from these impressions of the rate 0.824% unique click through rate per person or 0.073% clicks through per impressions.

Google analytics was used to track insights during the experiment in the period from April 21, 2015 to May 4, 2015. The total users participated in the experiment was 2784 users in 3260 sessions, they viewed 5410 pages on the website; the pages could be the homepage, the category page, or the article page, these details are visualized in Figure 3.

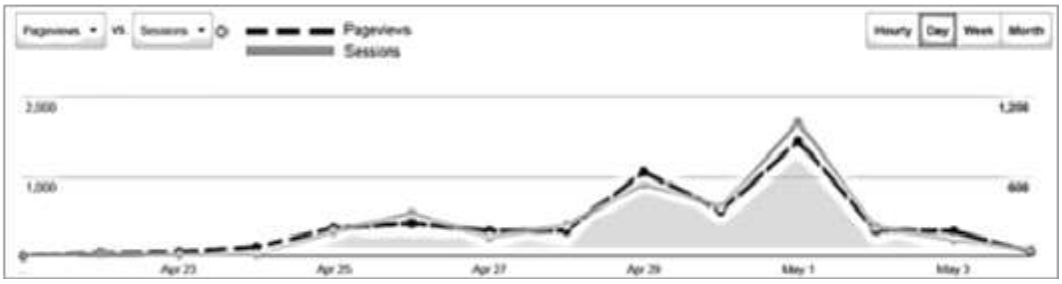

Figure 3. Pageviews Vs. Session During Experiment

Due to the lack of related studies about this field, the results were based on the same website as follow: The experiment used two intervals; the first interval was from April 21st, 2015 – April 27th, 2015 where the recommendation feature was not active on the website, the second interval was from April 28th, 2015 – May 4th, 2015 where the recommendation feature was active. For the ease of discussing of these periods; they will be referred to as “Recommendation Off” period, and “Recommendation On” period from now on. Figure 4 shows pageviews comparisons of both periods.

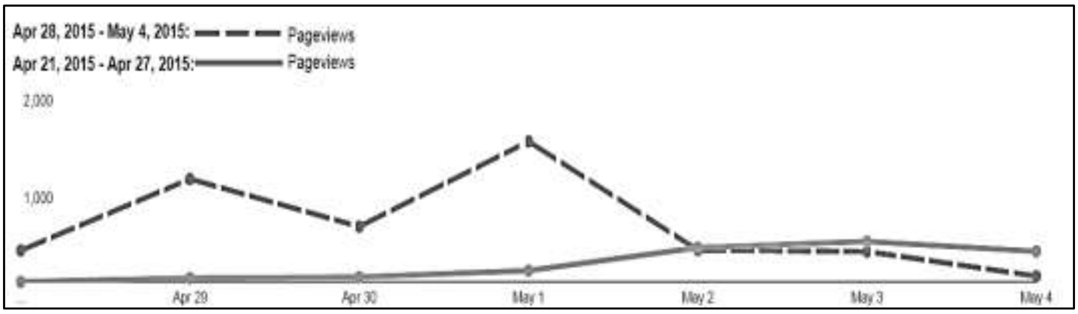

Figure 4. Comparing Experiments Intervals – Pageviews

Google defines Bounce Rate as “the percentage of single-page visits (i.e. visits in which the person left your site from the entrance page without interacting with the page)”. It is important to have clear definition for bounce rate to make sure that the performance numbers are correct, in all cases; the lower the bounce rate the better user’s engagement. Figure 5 illustrates the percentages of new users and returning based on Google bounce rate for each period.

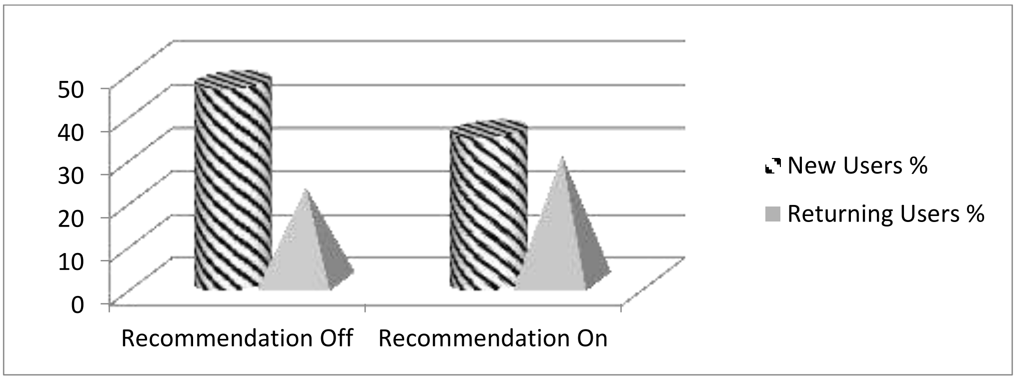

Figure 5. Google Bounce Rate Percentage for Experiment Periods

It is clear that the impact of the recommendation system from Figure 5 for google default bounce rate has dropped from 46.67% to 35.37% for new users with an improvement of 24.21% in bounce rate.

Although tracking was part of the whole website, this research is concerned about users’ behavior on article pages only, the results related to homepages and non-article pages (in which the recommendation feature was not part of the page) such as category pages or search results pages, will be omitted and only discuss numbers related to users’ behavior on the article pages. This is one of the reasons why it is advised to have a definition for the bounce to omit such pages and filter them out of the results. To achieve this goal, event tracking was used to record user interaction with article page elements, events were attached to defined elements that concern the targeted users of the recommendation system.

Figure 6 shows the defined elements and their locations, each element is targeted to measurement goal as follows:

i) Main Menu: this element is used to track users' clicks; any click on any menu item will be recorded as an event for Main Menu.

ii) Featured Sidebar: this element is used to track users' click; any click on any article in feature sidebar will be recorded as an event for featured sidebar.

iii) Article Start Area: this element is used to track users scroll to the beginning of the article area; once the page is loaded and the user scrolls to this area; an event will be recorded as scroll to article-start area.

iv) Recommendation Area: this element is used to track users scroll to the recommendation area; once the page is loaded and the user scrolls to this area; an event will be recorded as scroll to article-start area. Moreover, this area tracks clicks as well, any user clicks on any article in this area, an event will be triggered and recorded as recommendation area click.

v) Article End Area: this element is used to track users scroll to the end of the article; once the page is loaded and the user scrolls to the end of the article; an event will be recorded as scroll to article-start area.

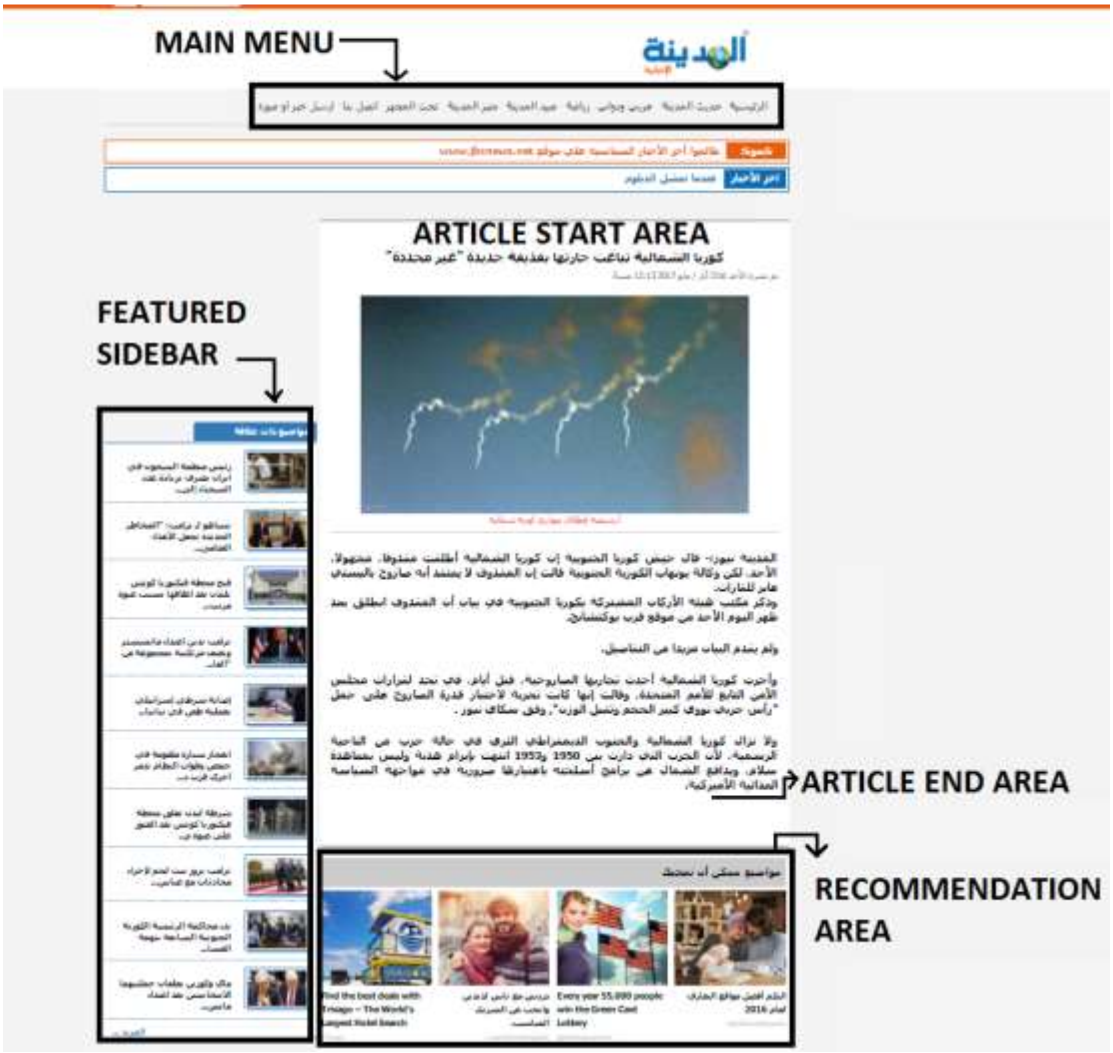

Figure 6. Article-Page Elements for Tracking Event

A goal is set for measuring users' interaction with recommendation area, this goal will record any user who uses clicks on any suggested article within the recommendation area. As mentioned earlier in this section; bounce rate definition will be changed to help us get better measurements of users' engagement on the article pages. We will filter users who interact with article page through events defined earlier. Only users who scrolled to the article-start area then scrolled to recommendation-area will be considered as potential users for the recommendation system, therefore, interactions of recommendations will be calculated based on people opened the page and scrolled down to the recommendation area. This number will be calculated by dividing the number of users engaged with recommendation system by the total number of users reached the recommendation area. This way results will be more accurate, only users who read the article and reached the recommendation area will be tracked. Moreover, this approach will help in indicating which articles are more user-engaging than other articles by tracking users at each part of the article and see how many users continue to the recommendation area or to the end of the article page without having any knowledge about the content; just the user's behavior.

After running the experiments for 2 weeks (first week with recommendation feature was off and the second week the recommendation feature was on), it has been noticed that the bounce rate is improved by 41.36%, it was dropped from 36.65% in the recommendation-off period, while it has been improved by 21.49% in the recommendation-on period. Users spent more time navigating website articles and pages, the average time on websites was improved by 39.45%, and it was changed from 00:03:23 in recommendation-off period to 00:04:43 in recommendation-on period.

Table 1. Users Behavior Drill Down Comparison

	Recommendation Off	Recommendation On
Total article page views	770	2640
Scroll to article start area	512	2094
Scroll to recommendation area	0	1798
Scroll to article end area	298	834
Click on featured sidebar	124	186
Click on main menu	38	126
Click on recommendation article	0	412

Table 1 shows the numbers used to compare the experiment periods; the total page views is 2640 page views, the first step was to filter users who just opened an article but did not read it, the article-start event was used to do so, only 2094 users started reading the article after opining it, that is almost 79.3% (2094/2640) of the total users who opened the article, this could happen due to many reasons, for example, the user might have opened the article page from a referral website in new tab and forgot to read it later. Second step was tracking users who continued to the end of the article and reached the recommendation area where article suggestions are showed, this indicator will help us to determine if the content is interesting to the users or if they got bored in middle of the content and closed the article page or move to another part of website. Also, 1798 users reached the recommendation area, in which it is 85.8% (1798/2094) of the total users started reading the article, there might be more users read the whole article but they did not scrolled to the recommendation area and moved to another part of the website or closed the browser tab after they finished, regardless of the reason; 412 users clicked on suggested articles from the recommendations suggested by the recommendation system, calculating the click-through-rate through dividing the number of clicks by the number of visits to find that 22.9% (412/1798) of users engaged with the recommendations and agreed that the suggestions were of their interest, this means that users' engagements from one article on the website have increased by 22.9% through the recommendation system. Figure 7. visualizes the drill down of users' behavior on article pages for the above experiment.

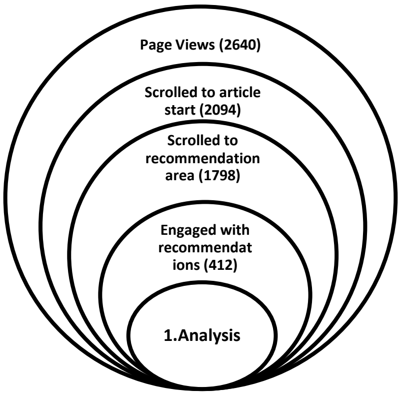
Page Views (2640)
Scrolled to article start (2094)
Scrolled to recommendation area (1798)
Engaged with recommendat ions (412)
1.Analysis

Figure 7. Drill Down of Users Behavior with Recommendation Enabled

4.4 Analysis

Increasing the engagement of users on a website content is very crucial as they are the base of the website; the test website has used industry methods to generate traffic of users to the website. The methods used were Online ads and social share. The Online ads method is without the recommendation feature and the social share method is with the recommendation feature, this way the impact of the recommendation system can be compared as a result of using the same website, the same targeted users, and the same content.

It has been found that users might load the article page but they did not interact with it at all, or they might start reading an article but left to another article or closed the article window before ending reading the article. These factors were considered, measured, and filtered from the results. Eventually, it has been found that 22.9% of active users interacted and engaged with the suggested articles on the recommendation area. Such engagement increases the website rating and increases the page views and time on website per user and reduced exits rate and bounce rate at the same time to make users surf more in the website and build loyalty base for returning and engaging users.

5. Conclusion and Future Work

Studies were concerned about recommendation systems for a long time, they studied the recommendation systems algorithms with offline techniques where recommendations are done in batches and served later. The field of online and real time recommendation systems is getting of interests for more researchers everyday as a result of the growth of both hardware and software at feasible budgets and as a result of cloud services offering resources as needed to suffice any kind of processing power.

This research had proposed a framework to build a real time recommendation system for news content websites. The framework used users' articles reading interests as rating input to generate recommendations and find similar items and users. Results of this recommendation framework showed 22.9% of users engagement with an increasing of 39.45% in average user time.

Other factors can be added implicitly to this recommendation framework in order to weight both input data and output recommendations such as considering the user's country, browser, technology, device, or considering the time of the day or connection speed as all of these factors can change user's behavior to narrow down similarity ranking between users. For example, users with high speed internet connection are more likely to play more video contents rather than users with low speed connections.

Future work can be done on other domains like social or e-commerce, mobile apps, and streaming services. A combined work can be done using the same framework and apply some user experience methods to the recommendation area and compare the engagement.

References

[1] http://www.worldwidewebsize.com/ (**2016**).

[2] Friedman, A., "Is the homepage dead?"-Columbia Journalism Review, http://www.cjr.org/realtalk/is_the_homepage_dead.php (**2015**).

[3] https://www.scribd.com/fullscreen/224608514 (**2015**).

[4] http://www.google.com/analytics/ (**2015**).

[5] Ltd., Last.fm: Last.fm (**2015**).

[6] Pandora Media, Inc.: Pandora® Internet Radio and the Music Genome Project® (**2015**).

[7] S. Abbar, S. Amer-Yahia, P. Indyk, and S. Mahabadi, "Real-time recommendation of diverse related articles", WWW '13, In Proceedings of the 22nd International Conference on World Wide Web, **(2013)**, Republic and Canton of Geneva, Switzerland, pp 1-12.

[8] A. Felfernig., M. Jeran, G. Ninaus, F. Reinfrank, and S. Reiterer, "Toward the Next Generation of Recommender Systems: Applications and Research Challenges", Multimedia Services in Intelligent Environments, Smart Innovation, Systems and Technologies, vol. 24, **(2013)**, pp. 81-98.

[9] J. Lu, S. Hoi, J. Wang, and P. Zhao, "Second Order Online Collaborative Filtering". The 5th Asian Conference on Machine Learning (ACML 2013), Canberra, Australia, **(2013)**, pp. 325-340.

[10] R. Suguna, and D. Sharmila, "An Efficient Web Recommendation System using Collaborative Filtering and Pattern Discovery Algorithms", International Journal of Computer Applications, (**2013)**, pp. 37-44.

[11] F. Ricci, L. Rokach, and B. Shapira, Introduction to Recommender Systems Handbook, Recommender Systems Handbook, Springer, **(2011)**, pp. 1-35.

[12] R. Lodha, H. Jain, and L. Kurup, "Big Data Challenges: Data Analysis Perspective", Computer Engineering Department, D.J.Sanghvi College of Engineering, vol. 4, no. 5, **(2014)**.

[13] McKinsey: Big Data: The next frontier for innovation, competition, and productivity, McKinsey Global Institute **(2011)**.

[14] A. Holzinger, C. Stocker, B. Ofner, G. Prohaska, A. Brabenetz, and R. Hofmann-Wellenhof, "Combining HCI, Natural Language Processing, and Knowledge Discovery - Potential of IBM Content Analytics as an Assistive Technology in the Biomedical Field". Holzinger, Andreas; Pasi, Gabriella. Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data, **(2013)**, pp 13–24.

[15] A. Said, S. Dooms, B. Loni, and D. Tikk, "Recommender systems challenge 2014", RecSys '14, Proceedings of the 8th ACM Conference on Recommender systems, **(2014)**, pp. 387-388.

[16] https://wordpress.org/ (**2015**).

[17] http://php.net/ (**2015**).

[18] https://mariadb.org/ (**2015**).

[19] http://nginx.org/ (**2015**).

[20] http://lucene.apache.org/solr/ (**2015**).

[21] http://www.solarium-project.org/ (**2015**).

[22] https://www.mongodb.org/ (**2015**).

[23] http://www.hpl.hp.com/research/linux/httperf/ (**2015**).

[24] https://www.online.net/en/dedicated-server/dedibox-pro (**2015**).